LLMs: Multimodal Models

Lecture 16

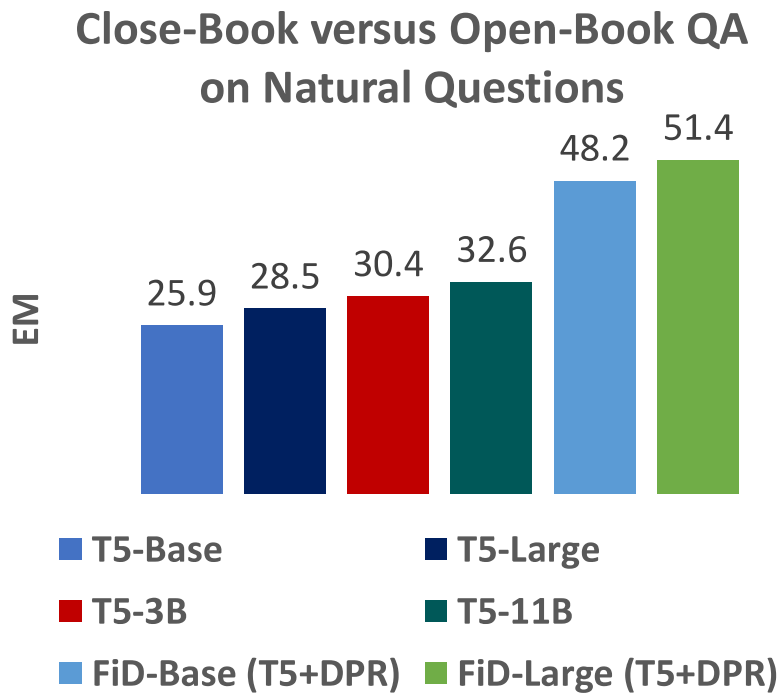
18-789

Administrative

- Today
  - RAG
  - Multimodal Models

Retrieval-Augmented Pretraining

LLMs can memorize lots of knowledge in its parameters

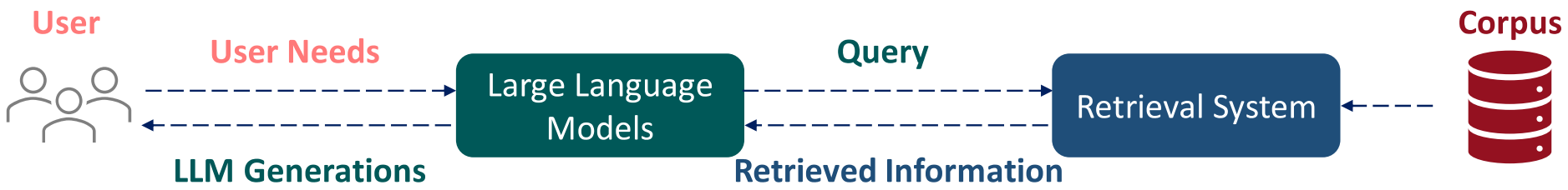


Clear benefits from adding external information from a retrieval system on knowledge-intensive tasks

- Better overall accuracy
- Significantly better parameter efficiency
- More reliable/explicit source of information

Retrieval Augmentation

- Can we alleviate LLMs from costly parametric memory by augmenting them with a retrieval system?
  - Retrieval system serves as an external memory
  - LLMs learn to leverage retrieved information
- Ideally:
  - More efficient parameter usage
  - Better generalization ability by switching external memory (retrieval corpus)
  - Clear separation of memorization and understanding for transparency and control



KNN-LM

GENERALIZATION THROUGH MEMORIZATION: NEAREST NEIGHBOR LANGUAGE MODELS

Urvashi Khandelwal[†]*, **Omer Levy**[‡], **Dan Jurafsky**[†], **Luke Zettlemoyer**[‡] **& Mike Lewis**[‡]
[†]Stanford University
[‡]Facebook AI Research
{urvashik,jurafsky}@stanford.edu
{omerlevy,lsz,mikelewis}@fb.com

ABSTRACT

We introduce kNN-LMs, which extend a pre-trained neural language model (LM) by linearly interpolating it with a k-nearest neighbors (kNN) model. The nearest neighbors are computed according to distance in the pre-trained LM embedding space, and can be drawn from any text collection, including the original LM training data. Applying this augmentation to a strong WIKITEXT-103 LM, with neighbors drawn from the original training set, our kNN-LM achieves a new state-of-the-art perplexity of 15.79 – a 2.9 point improvement with no additional training. We also show that this approach has implications for efficiently scaling up to larger training sets and allows for effective domain adaptation, by simply varying the nearest neighbor datastore, again without further training. Qualitatively, the model is particularly helpful in predicting rare patterns, such as factual knowledge. Together, these results strongly suggest that learning similarity between sequences of text is easier than predicting the next word, and that nearest neighbor search is an effective approach for language modeling in the long tail.

1 INTRODUCTION

Neural language models (LMs) typically solve two subproblems: (1) mapping sentence prefixes to fixed-sized representations, and (2) using these representations to predict the next word in the text (Bengio et al., 2003; Mikolov et al., 2010). We present a new language modeling approach that is based on the hypothesis that the representation learning problem may be easier than the prediction problem. For example, any English speaker knows that *Dickens is the author of* and *Dickens wrote* will have essentially the same distribution over the next word, even if they do not know what that distribution is. We provide strong evidence that existing language models, similarly, are much better at the first problem, by using their prefix embeddings in a simple nearest neighbor scheme that significantly improves overall performance.

KNN-LM: Interpolate LM prediction with a k-nearest neighbor (KNN) model

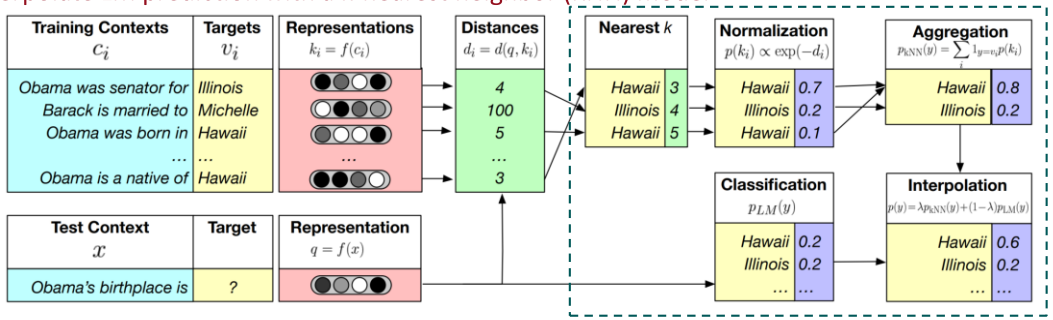


Figure 1: The pipeline of KNN-LM [2]

When to retrieve?

- Every token position at inference time

What to retrieve?

- (Context, Target Word) pairs: (c_i, v_i)
- Key: $k = f(c_i)$ the hidden representation of context from the LM
- Value: the actual ground truth word for the context

KNN-LM

GENERALIZATION THROUGH MEMORIZATION: NEAREST NEIGHBOR LANGUAGE MODELS

Urvashi Khandelwal[†]*, **Omer Levy**[‡], **Dan Jurafsky**[†], **Luke Zettlemoyer**[‡] **& Mike Lewis**[‡]
[†]Stanford University
[‡]Facebook AI Research
{urvashik,jurafsky}@stanford.edu
{omerlevy,lsz,mikelewis}@fb.com

ABSTRACT

We introduce kNN-LMs, which extend a pre-trained neural language model (LM) by linearly interpolating it with a k-nearest neighbors (kNN) model. The nearest neighbors are computed according to distance in the pre-trained LM embedding space, and can be drawn from any text collection, including the original LM training data. Applying this augmentation to a strong WIKITEXT-103 LM, with neighbors drawn from the original training set, our kNN-LM achieves a new state-of-the-art perplexity of 15.79 – a 2.9 point improvement with no additional training. We also show that this approach has implications for efficiently scaling up to larger training sets and allows for effective domain adaptation, by simply varying the nearest neighbor datastore, again without further training. Qualitatively, the model is particularly helpful in predicting rare patterns, such as factual knowledge. Together, these results strongly suggest that learning similarity between sequences of text is easier than predicting the next word, and that nearest neighbor search is an effective approach for language modeling in the long tail.

1 INTRODUCTION

Neural language models (LMs) typically solve two subproblems: (1) mapping sentence prefixes to fixed-sized representations, and (2) using these representations to predict the next word in the text (Bengio et al., 2003; Mikolov et al., 2010). We present a new language modeling approach that is based on the hypothesis that the representation learning problem may be easier than the prediction problem. For example, any English speaker knows that *Dickens is the author of* and *Dickens wrote* will have essentially the same distribution over the next word, even if they do not know what that distribution is. We provide strong evidence that existing language models, similarly, are much better at the first problem, by using their prefix embeddings in a simple nearest neighbor scheme that significantly improves overall performance.

KNN-LM: Interpolate LM prediction with a k-nearest neighbor (KNN) model

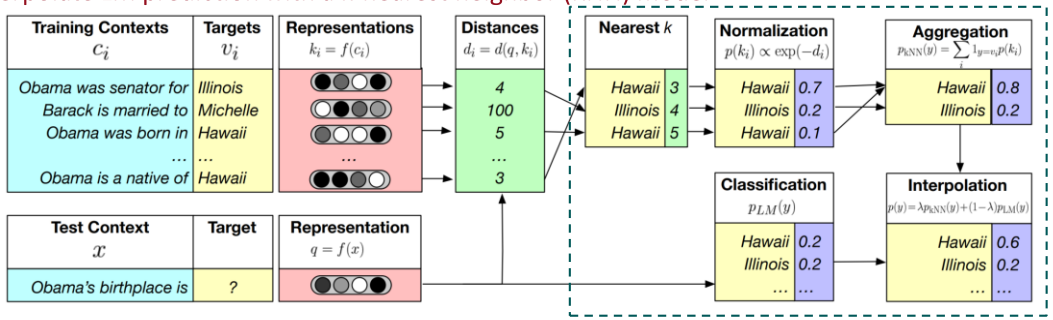


Figure 1: The pipeline of KNN-LM [2]

How to retrieve?

- K nearest neighbor search using current context x
- Query: $q = f(x)$ the hidden representation of the inference context
- Retrieval Function: standard KNN search with L2 distance

Where to retrieve?

- The training corpus with the pretrained LM to formulate the key-value pairs
- Or plug-in a new corpus to generalize to the new domain. E.g. Wiki→Book

KNN-LM

GENERALIZATION THROUGH MEMORIZATION: NEAREST NEIGHBOR LANGUAGE MODELS

Urvashi Khandelwal[†,*], **Omer Levy**[‡], **Dan Jurafsky**[†], **Luke Zettlemoyer**[‡] **& Mike Lewis**[‡]
[†]Stanford University
[‡]Facebook AI Research
{urvashik,jurafsky}@stanford.edu
{omerlevy,lsz,mikelewis}@fb.com

ABSTRACT

We introduce kNN-LMs, which extend a pre-trained neural language model (LM) by linearly interpolating it with a k-nearest neighbors (kNN) model. The nearest neighbors are computed according to distance in the pre-trained LM embedding space, and can be drawn from any text collection, including the original LM training data. Applying this augmentation to a strong WIKITEXT-103 LM, with neighbors drawn from the original training set, our kNN-LM achieves a new state-of-the-art perplexity of 15.79 – a 2.9 point improvement with no additional training. We also show that this approach has implications for efficiently scaling up to larger training sets and allows for effective domain adaptation, by simply varying the nearest neighbor datastore, again without further training. Qualitatively, the model is particularly helpful in predicting rare patterns, such as factual knowledge. Together, these results strongly suggest that learning similarity between sequences of text is easier than predicting the next word, and that nearest neighbor search is an effective approach for language modeling in the long tail.

1 INTRODUCTION

Neural language models (LMs) typically solve two subproblems: (1) mapping sentence prefixes to fixed-sized representations, and (2) using these representations to predict the next word in the text (Bengio et al., 2003; Mikolov et al., 2010). We present a new language modeling approach that is based on the hypothesis that the representation learning problem may be easier than the prediction problem. For example, any English speaker knows that *Dickens is the author of* and *Dickens wrote* will have essentially the same distribution over the next word, even if they do not know what that distribution is. We provide strong evidence that existing language models, similarly, are much better at the first problem, by using their prefix embeddings in a simple nearest neighbor scheme that significantly improves overall performance.

KNN-LM: Interpolate LM prediction with a k-nearest neighbor (KNN) model

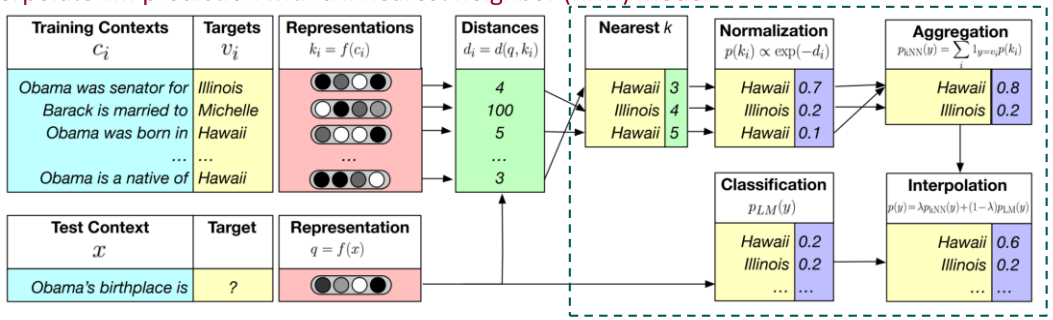


Figure 1: The pipeline of KNN-LM [2]

How to leverage retrieved information?

- Normalize and aggregate retrieved scores to obtain the KNN output
- Linearly interpolate the original LM output with KNN output

$$p(y) = \lambda p_{\text{knn}}(y) + (1-\lambda) p_{\text{lm}}(y)$$

REALM

REALM: Retrieval-Augmented Language Model Pre-Training

Kelvin Guu [*1] Kenton Lee [*1] Zora Tung [1] Panupong Pasupat [1] Ming-Wei Chang [1]

Abstract

Language model pre-training has been shown to capture a surprising amount of world knowledge, crucial for NLP tasks such as question answering. However, this knowledge is stored implicitly in the parameters of a neural network, requiring ever-larger networks to cover more facts. To capture knowledge in a more modular and interpretable way, we augment language model pre-training with a latent knowledge retriever, which allows the model to retrieve and attend over documents from a large corpus such as Wikipedia, used during pre-training, fine-tuning and inference. For the first time, we show how to pre-train such a knowledge retriever in an unsupervised manner, using masked language modeling as the learning signal and backpropagating through a retrieval step that considers millions of documents. We demonstrate the effectiveness of Retrieval-Augmented Language Model pre-training (REALM) by fine-tuning on the challenging task of Open-domain Question Answering (Open-QA). We compare against state-of-the-art models for both explicit and implicit knowledge storage on three popular Open-QA benchmarks, and find that we outperform all previous methods by a significant margin (4-16% absolute accuracy), while also providing qualitative benefits such as interpretability and modularity.

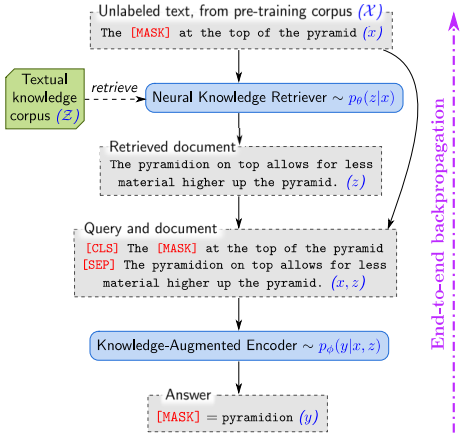


Figure 1. REALM augments language model pre-training with a neural knowledge retriever that retrieves knowledge from a textual knowledge corpus, Z (e.g., all of Wikipedia). Signal from the language modeling objective backpropagates all the way through the retriever, which must consider millions of documents in Z—a significant computational challenge that we address.

correctly predict the missing word in the following sentence: "The ___ is the currency of the United Kingdom" (answer: "pound").

When to retrieve?
- Once every training/testing sequence

What to retrieve?
- Similar text sequences

How to retrieve?
- Dense retriever (BERT here) using current sequence as the query

Where to retrieve?
- The same pretraining corpus

How to leverage retrieved information?
- Add retrieved sequence as extra inputs
- Pretrain LM to learn how to use these extra information (hopefully)

Retro

Improving language models by retrieving from trillions of tokens

Sebastian Borgeaud†, Arthur Mensch†, Jordan Hoffmann†, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae‡, Erich Elsen‡ and Laurent Sifre†,‡

All authors from DeepMind, †Equal contributions, ‡Equal senior authorship

We enhance auto-regressive language models by conditioning on document chunks retrieved from a large corpus, based on local similarity with preceding tokens. With a 2 trillion token database, our Retrieval-Enhanced Transformer (Ret r o) obtains comparable performance to GPT-3 and Jurassic-1 on the Pile, despite using 25→ fewer parameters. After fine-tuning, Ret r o performance translates to downstream knowledge-intensive tasks such as question answering. Ret r o combines a frozen Ber t retriever, a differentiable encoder and a chunked cross-attention mechanism to predict tokens based on an order of magnitude more data than what is typically consumed during training. We typically train Ret r o from scratch, yet can also rapidly Ret r ofit pre-trained transformers with retrieval and still achieve good performance. Our work opens up new avenues for improving language models through explicit memory at unprecedented scale.

1. Introduction

Language modelling (LM) is an unsupervised task that consists of modelling the probability of text, usually by factorising it into conditional next-token predictions $>(F_1, \ldots, F_<) = \prod_7 >(F_7|F_{<7})$. Neural networks have proven to be powerful language models, first in the form of recurrent architectures (Graves, 2013; Jozefowicz et al., 2016; Mikolov et al., 2010) and more recently in the form of Transformers (Vaswani et al., 2017), that use attention to contextualise the past. Large performance improvements have come from increasing the amount of data, training compute, or model parameters. Transformers have been scaled from 100 million parameter models in seminal work to over hundred billion parameters (Brown et al., 2020; Radford et al., 2019) in the last two years which has led to models that do very well on a wide array of tasks in a zero or few-shot formulation. Increasing model

When to retrieve?
- Split a pretraining sequence into chunks (e.g. 64 tokens per chunk)
- Retrieve similar chunks for each chunk

What to retrieve?
- Forming key-value pairs as (this chunk, next chunk) from documents of the corpus
- Retrieve key chunks (x), augment value chunk (y).

How to retrieve?
- Represent chunks by average BERT embeddings across its token positions
- Retrieval by L2 distance in their representations from ANNS index
- Use SCANN to enable 10 million second latency per querying from 2 trillion tokens (31 Billion Embeddings)

Where to retrieve?
- All chunks from the pretraining corpus, embedded by frozen BERTs
- Again, can switch the retrieval corpus at inference time

How to leverage retrieved information
- New Chunked Cross-Attention (CCA) blocks: each chunk attends to retrieved chunks of the previous chunk
- Inter leaving CCA blocks and normal decoder attention blocks in each RETRO attention layer
- Pretraining end-to-end at large scale

Memory3

Memory^3: Language Modeling with Explicit Memory

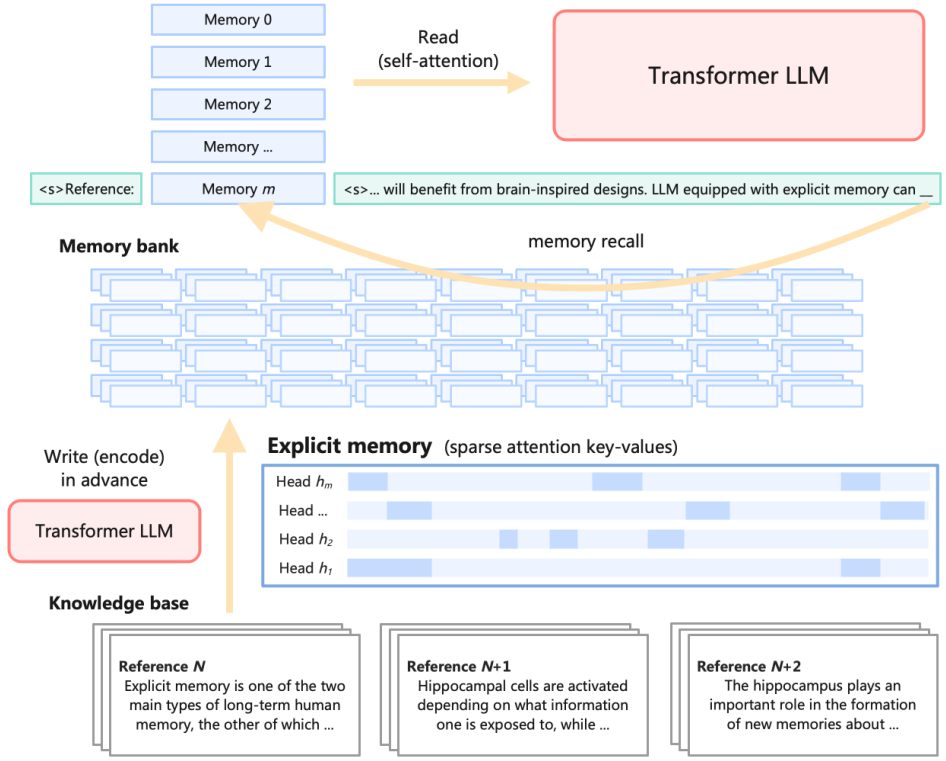
Hongkang Yang[1], Zehao Lin[1], Wenjin Wang[1], Hao Wu[1], Zhiyu Li[1], Bo Tang[1], Wenqiang Wei[1], Jinbo Wang[1], Zeyun Tang[1], Shichao Song[1], Chenyang Xi[1], Yu Yu[1], Kai Chen[1], Feiyu Xiong[1], Linpeng Tang[2], and Weinan E[*3,1]

[1]Center for LLM, Institute for Advanced Algorithms Research, Shanghai
[2]Moqi Inc
[3]Center for Machine Learning Research, Peking University

July 1, 2024

Abstract

The training and inference of large language models (LLMs) are together a costly process that transports knowledge from raw data to meaningful computation. Inspired by the memory hierarchy of the human brain, we reduce this cost by equipping LLMs with explicit memory, a memory format cheaper than model parameters and text retrieval-augmented generation (RAG). Conceptually, with most of its knowledge externalized to explicit memories, the LLM can enjoy a smaller parameter size, training cost, and inference cost, all proportional to the amount of remaining "abstract knowledge". As a preliminary proof of concept, we train from scratch a 2.4B LLM, which achieves better performance than much larger LLMs as well as RAG models, and maintains higher decoding speed than RAG. The model is named Memory^3, since explicit memory is the third form of memory in LLMs after implicit memory (model parameters) and working memory (context key-values). We introduce a memory circuitry theory to support the externalization of knowledge, and present novel techniques including a memory sparsification mechanism that makes storage tractable and a two-stage pretraining scheme that facilitates memory formation.



Retrieve In-context Examples

Learning To Retrieve Prompts for In-Context Learning

Ohad Rubin **Jonathan Herzig** **Jonathan Berant**
The Blavatnik School of Computer Science, Tel Aviv University
{ohad.rubin,jonathan.herzig,joberant}@cs.tau.ac.il

Abstract

In-context learning is a recent paradigm in natural language understanding, where a large pre-trained language model (LM) observes a test instance and a few training examples as its input, and directly decodes the output without any update to its parameters. However, performance has been shown to strongly depend on the selected training examples (termed *prompts*). In this work, we propose an efficient method for retrieving prompts for in-context learning using annotated data and an LM. Given an input-output pair, we estimate the probability of the output given the input and a candidate training example as the prompt, and label training examples as positive or negative based on this probability. We then train an efficient dense retriever from this data, which is used to retrieve training examples as prompts at test time. We evaluate our approach on three sequence-to-sequence tasks where language utterances are mapped to meaning representations, and find that it sub-

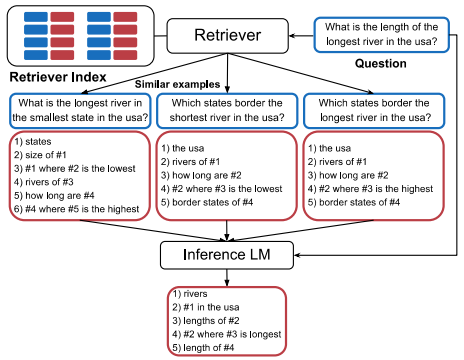


Figure 1: An overview of prompt retrieval: Given a question from BREAK, one retrieves similar training examples from an index of the training set. The question and training examples (the prompt) are passed to an inference LM that decodes the output.

(2021a) showed that downstream performance can

When to retrieve?
- Per downstream data point

What to retrieve?
- Similar training data points (x, y)

How to retrieve?
- Dense retriever adapted for LLM

Where to retrieve?
- Training data

How to leverage retrieved information?
- Add in as in-context examlpes

Retrieve In-context Examples

Learning To Retrieve Prompts for In-Context Learning

Ohad Rubin **Jonathan Herzig** **Jonathan Berant**
The Blavatnik School of Computer Science, Tel Aviv University
{ohad.rubin,jonathan.herzig,joberant}@cs.tau.ac.il

Abstract

In-context learning is a recent paradigm in natural language understanding, where a large pre-trained language model (LM) observes a test instance and a few training examples as its input, and directly decodes the output without any update to its parameters. However, performance has been shown to strongly depend on the selected training examples (termed *prompts*). In this work, we propose an efficient method for retrieving prompts for in-context learning using annotated data and an LM. Given an input-output pair, we estimate the probability of the output given the input and a candidate training example as the prompt, and label training examples as positive or negative based on this probability. We then train an efficient dense retriever from this data, which is used to retrieve training examples as prompts at test time. We evaluate our approach on three sequence-to-sequence tasks where language utterances are mapped to meaning representations, and find that it sub-

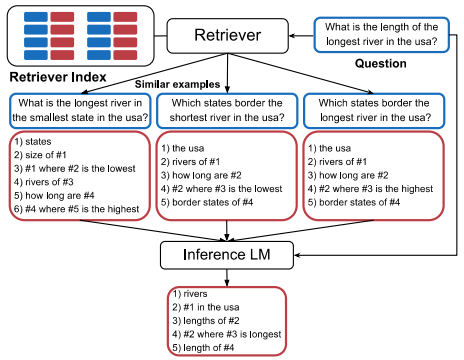


Figure 1: An overview of prompt retrieval: Given a question from BREAK, one retrieves similar training examples from an index of the training set. The question and training examples (the prompt) are passed to an inference LM that decodes the output.

(2021a) showed that downstream performance can

When does this work?

- A generally better way to fine more similar in-context examples than random sample

Why does this work?

- A form of test time learning

Test Time Learning

- Find similar training data points for the current testing data point
- Focused learning (e.g., a few gradient steps) on similar training data points
- "Upweighting" training data close to the current testing data
- A classic idea

"In-context learning == SGD view"

- Performing virtual SGD on random versus retrieved similar in-context examples

Retrieve Extra Knowledge

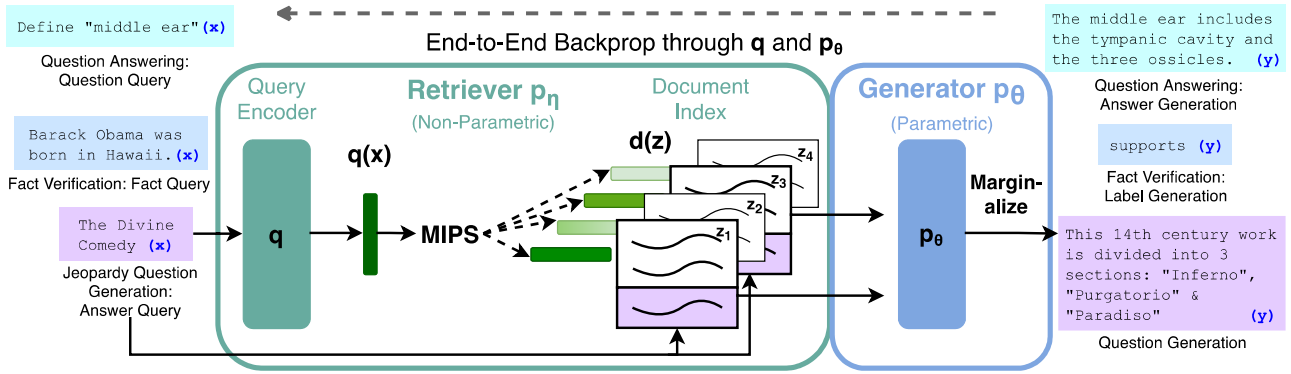
Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis[†‡], **Ethan Perez**[⋆], **Aleksandra Piktus**[†], **Fabio Petroni**[†], **Vladimir Karpukhin**[†], **Naman Goyal**[†], **Heinrich Küttler**[†], **Mike Lewis**[†], **Wen-tau Yih**[†], **Tim Rocktäschel**[†‡], **Sebastian Riedel**[†‡], **Douwe Kiela**[†]

[†]Facebook AI Research; [‡]University College London; [⋆]New York University;
plewis@fb.com

Abstract

Large pre-trained language models have been shown to store factual knowledge in their parameters, and achieve state-of-the-art results when fine-tuned on downstream NLP tasks. However, their ability to access and precisely manipulate knowledge is still limited, and hence on knowledge-intensive tasks, their performance lags behind task-specific architectures. Additionally, providing provenance for their decisions and updating their world knowledge remain open research problems. Pre-trained models with a differentiable access mechanism to explicit non-parametric memory have so far been only investigated for extractive downstream tasks. We explore a general-purpose fine-tuning recipe for retrieval-augmented generation (RAG) — models which combine pre-trained parametric and non-parametric memory for language generation. We introduce RAG models where the parametric memory is a pre-trained seq2seq model and the non-parametric memory is a dense vector index of Wikipedia, accessed with a pre-trained neural retriever. We compare two RAG formulations, one which conditions on the same retrieved passages across the whole generated sequence, and another which can use different passages



When to retrieve?
- Per downstream data point

What to retrieve?
- Relevant documents (x)

How to retrieve?
- Dense retriever, e.g., from web search

Where to retrieve?
- Target corpus with needed information

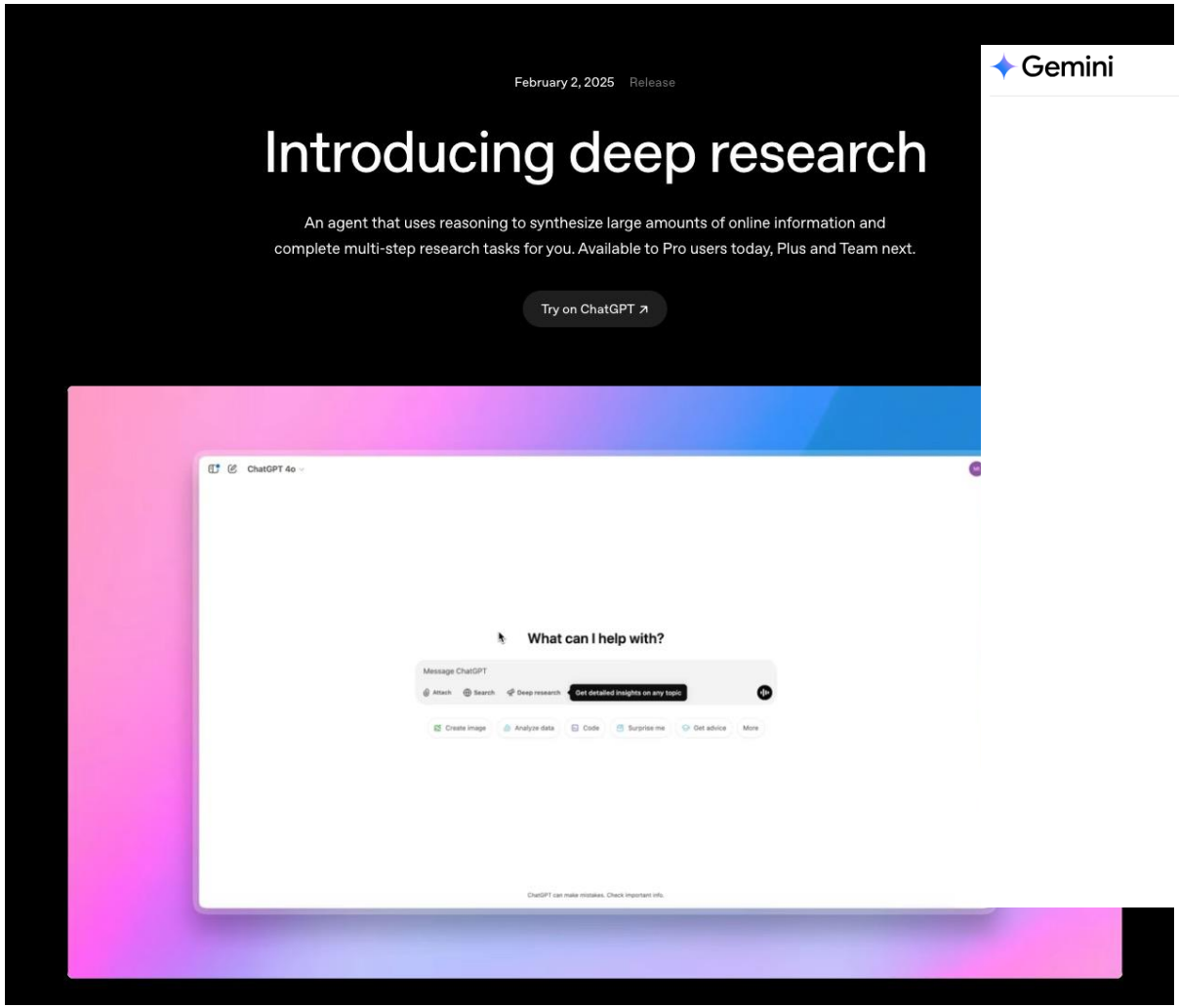
How to leverage retrieved information?
- Used to be complex:
  - Latent space models
  - Fusion-in-Decoder
- With Decoder LLM:
  - As additional inputs (with prompts)
  - Zero-shot or finetuned

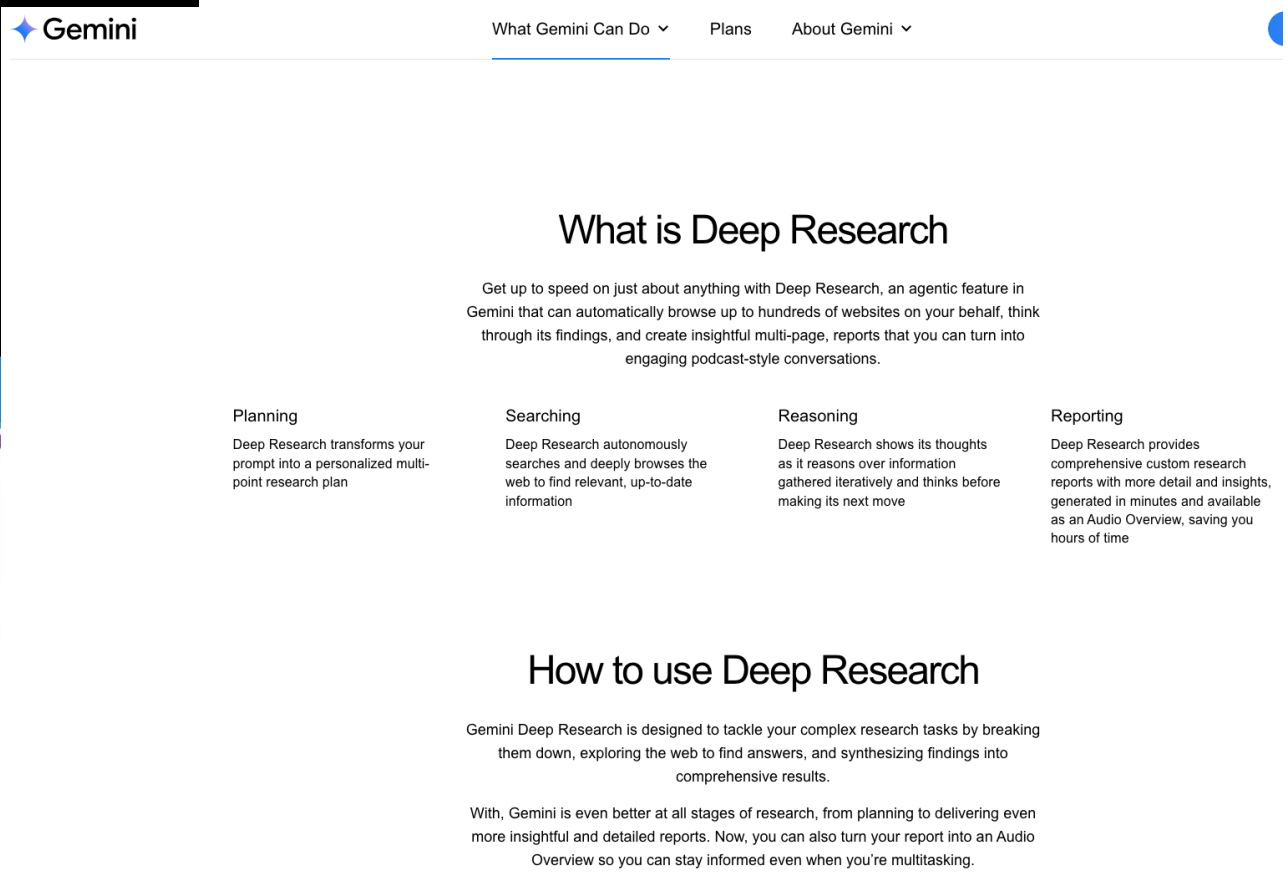
When does this work?
- Tasks required additional information
- E.g., Knowledge-intensive tasks, reducing hallucination, plug-in in-domain information, etc.

Why does it work?
- Additional information from retrieval
- An LLM version of OpenQA

When does it **not** work?
- Tasks do not require extra information
- E.g., hard to think of what to extra information is needed for sentiment analysis or grammar check

Deep Research





Multi-Modal Models

- Models that can incorporate *multiple* modalities
  - Images
  - Video
  - Text
  - Audio
  - …
- Ideal Outcomes: (1) achieve multimodal understanding, (2) achieve better *unimodal* understanding

Credit to UC Berkeley Deep Unsupervised Learning

Types

- **Bidirectional Multi-Modal Models**
- Encoder-Decoder Multi-Modal Models
- Decoder-only Multi-Modal Models
- Multi-Modal Generation

CLIP

Learning Transferable Visual Models From Natural Language Supervision

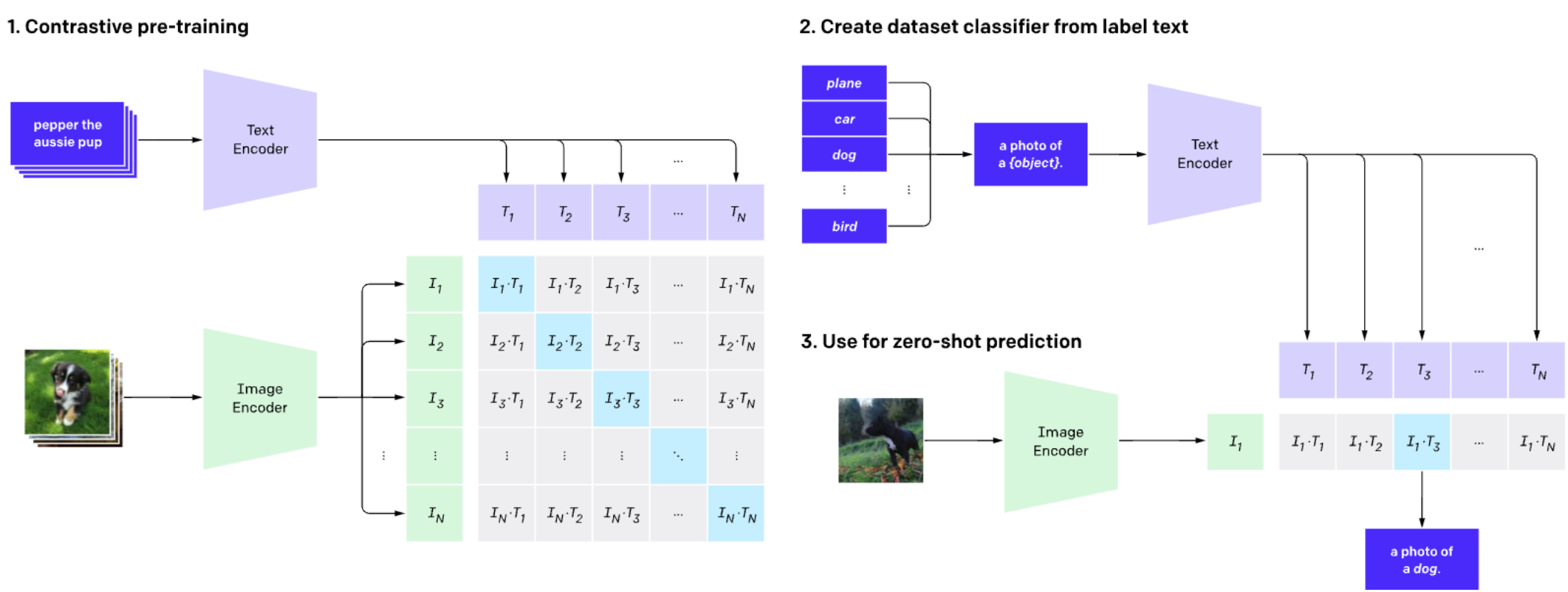
Alec Radford [*1] Jong Wook Kim [*1] Chris Hallacy [1] Aditya Ramesh [1] Gabriel Goh [1] Sandhini Agarwal [1]
Girish Sastry [1] Amanda Askell [1] Pamela Mishkin [1] Jack Clark [1] Gretchen Krueger [1] Ilya Sutskever [1]

Dataset

- Existing text annotated image dataset at the time were relatively small
- YFCC100M
  - Text metadata quality is low, some captions are automatically generated file names like “20160716 113957.JPG”
- Constructed dataset of 400M image-text pairs
- Images searched with one of 500K generated queries

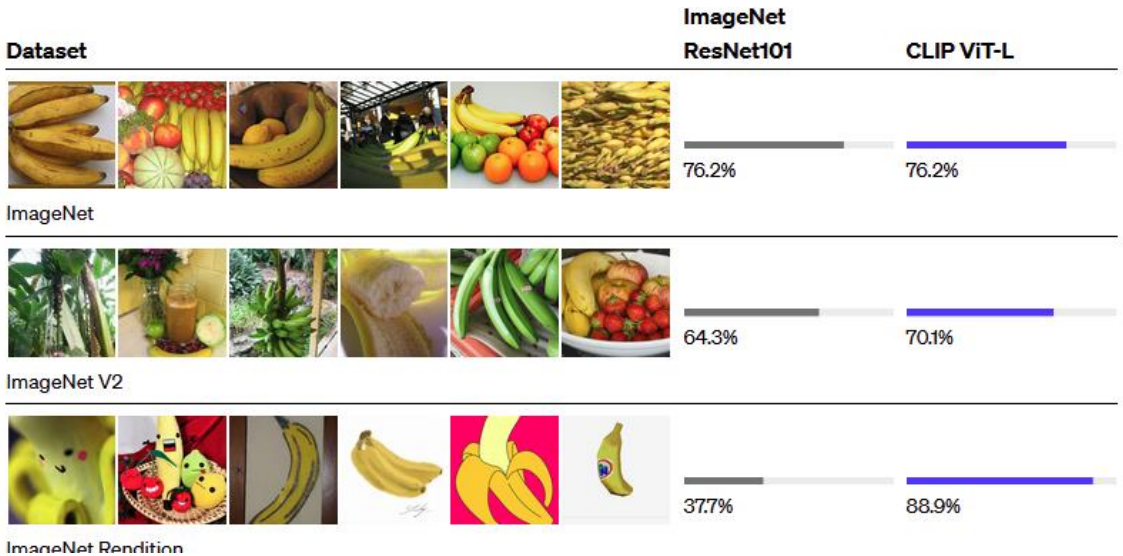
```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

CLIP


1. Contrastive pre-training
pepper the aussie pup
Text Encoder
Image Encoder
2. Create dataset classifier from label text
plane
car
dog
bird
a photo of a {object}.
Text Encoder
3. Use for zero-shot prediction
Image Encoder
a photo of a dog.

CLIP: Capabilities

Image classification



CLIP score as a re-ranker

- Generation N images from a text-image model
- Rank their text-image alignment using CLIP
- Select the pair with the highest CLIP score

CLIP: Capabilities

Image generation (guidance) - VQGAN CLIP

3D generation (Dream Fields)

CLIP: Capabilities

Use CLIP text-image encoders as pretrained models

- Many current vision-language models
- For representations (e.g. Stable Diffusion text encoder)
- CLIPort

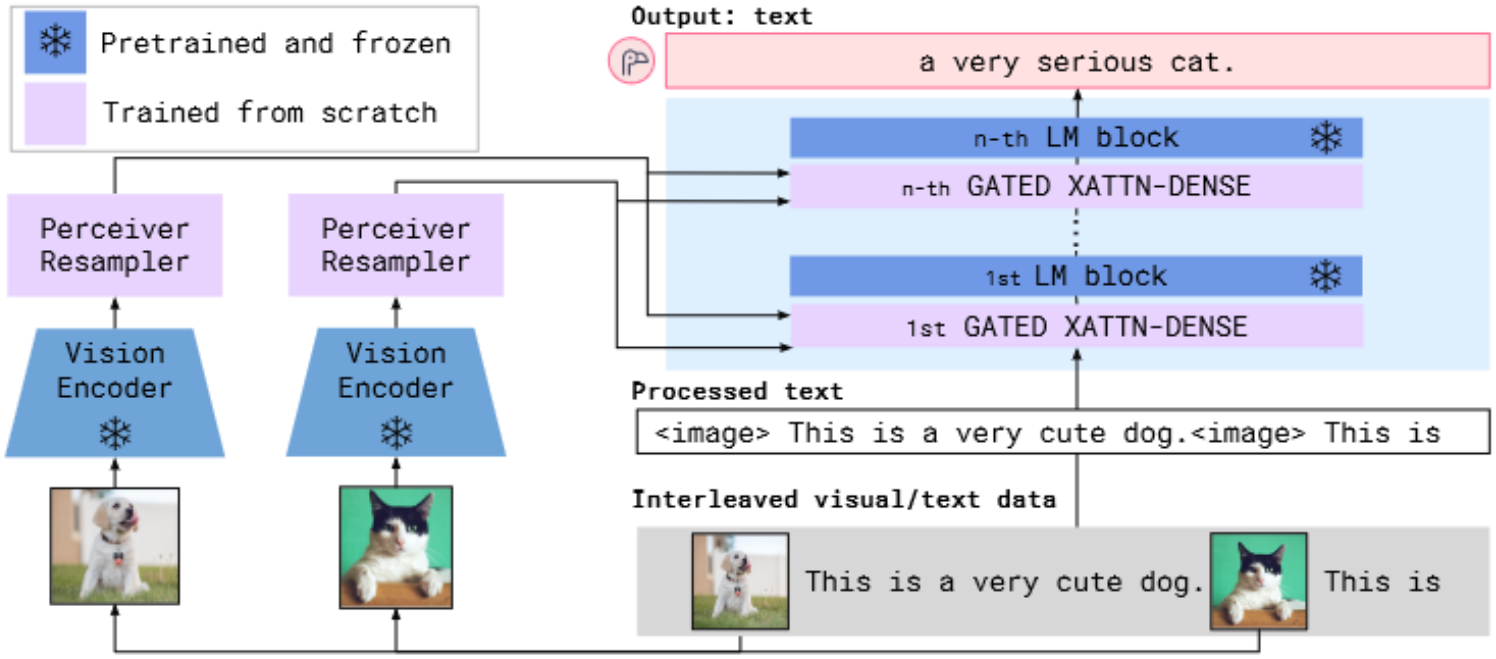
Usually achieves better generalization, especially to unseen visual concepts during finetuning

Types

- Bidirectional Multi-Modal Models
- **Encoder-Decoder Multi-Modal Models**
- Decoder-only Multi-Modal Models
- Multi-Modal Generation

Flamingo

Leveraging pretrained vision-only and language-only models



Flamingo: a Visual Language Model for Few-Shot Learning

Jean-Baptiste Alayrac*,‡ Jeff Donahue* Pauline Luc* Antoine Miech*
Iain Barr† Yana Hasson† Karel Lenc† Arthur Mensch† Katie Millican†
Malcolm Reynolds† Roman Ring† Eliza Rutherford† Serkan Cabi Tengda Han
Zhitao Gong Sina Samangooei Marianne Monteiro Jacob Menick
Sebastian Borgeaud Andrew Brock Aida Nematzadeh Sahand Sharifzadeh
Mikolaj Binkowski Ricardo Barreira Oriol Vinyals Andrew Zisserman
Karen Simonyan*,‡

* Equal contributions, ordered alphabetically, † Equal contributions, ordered alphabetically, ‡ Equal senior contributions

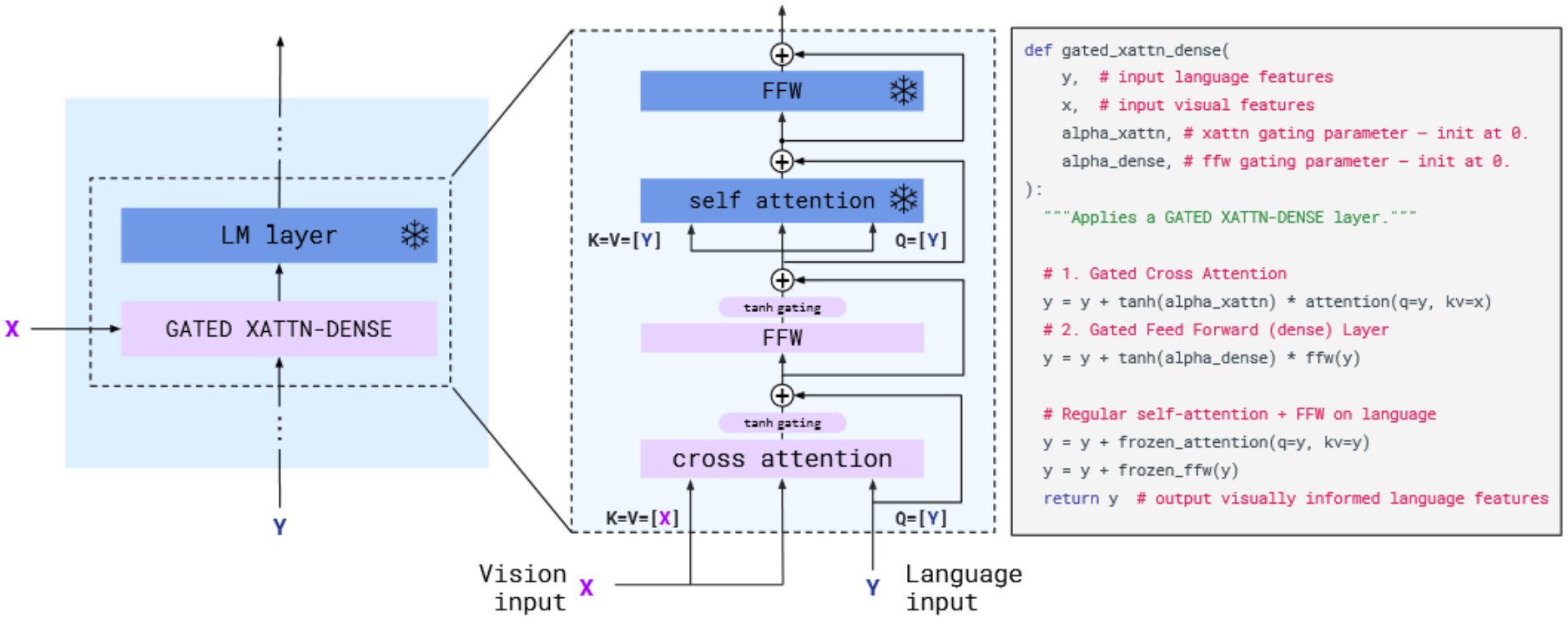
DeepMind

Abstract

Building models that can be rapidly adapted to novel tasks using only a handful of annotated examples is an open challenge for multimodal machine learning research. We introduce Flamingo, a family of Visual Language Models (VLM) with this ability. We propose key architectural innovations to: (i) bridge powerful pretrained vision-only and language-only models, (ii) handle sequences of arbitrarily interleaved visual and textual data, and (iii) seamlessly ingest images or videos as inputs. Thanks to their flexibility, Flamingo models can be trained on large-scale multimodal web corpora containing arbitrarily interleaved text and images, which is key to endow them with in-context few-shot learning capabilities. We perform a thorough evaluation of our models, exploring and measuring their ability to rapidly adapt to a variety of image and video tasks. These include open-ended tasks such as visual question-answering, where the model is prompted with a question which it has to answer; captioning tasks, which evaluate the ability to describe a scene or an event; and close-ended tasks such as multiple-choice visual question-answering. For tasks lying anywhere on this spectrum, a *single* Flamingo model can achieve a new state of the art with few-shot learning, simply by prompting the model with task-specific examples. On numerous benchmarks, *Flamingo* outperforms models fine-tuned on thousands of times more task-specific data.

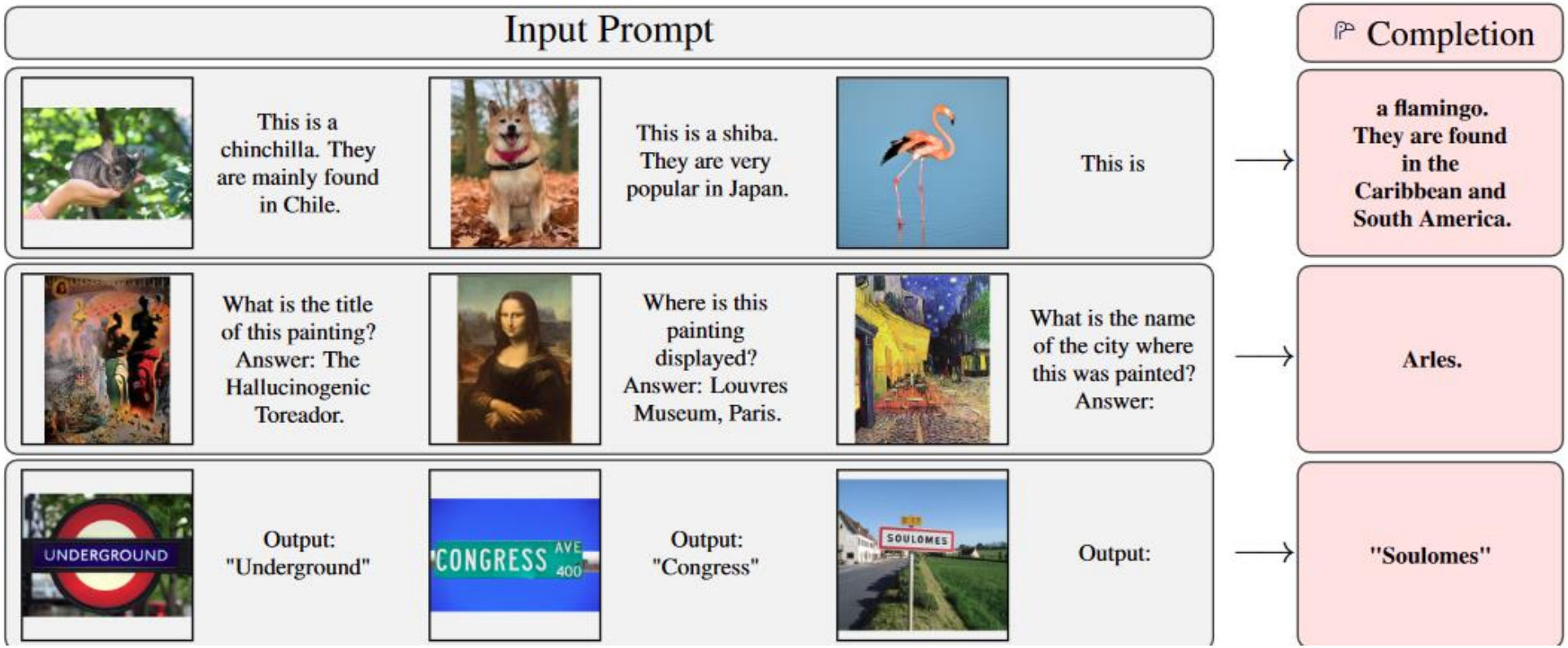
Chinchilla + CLIP

Flamingo

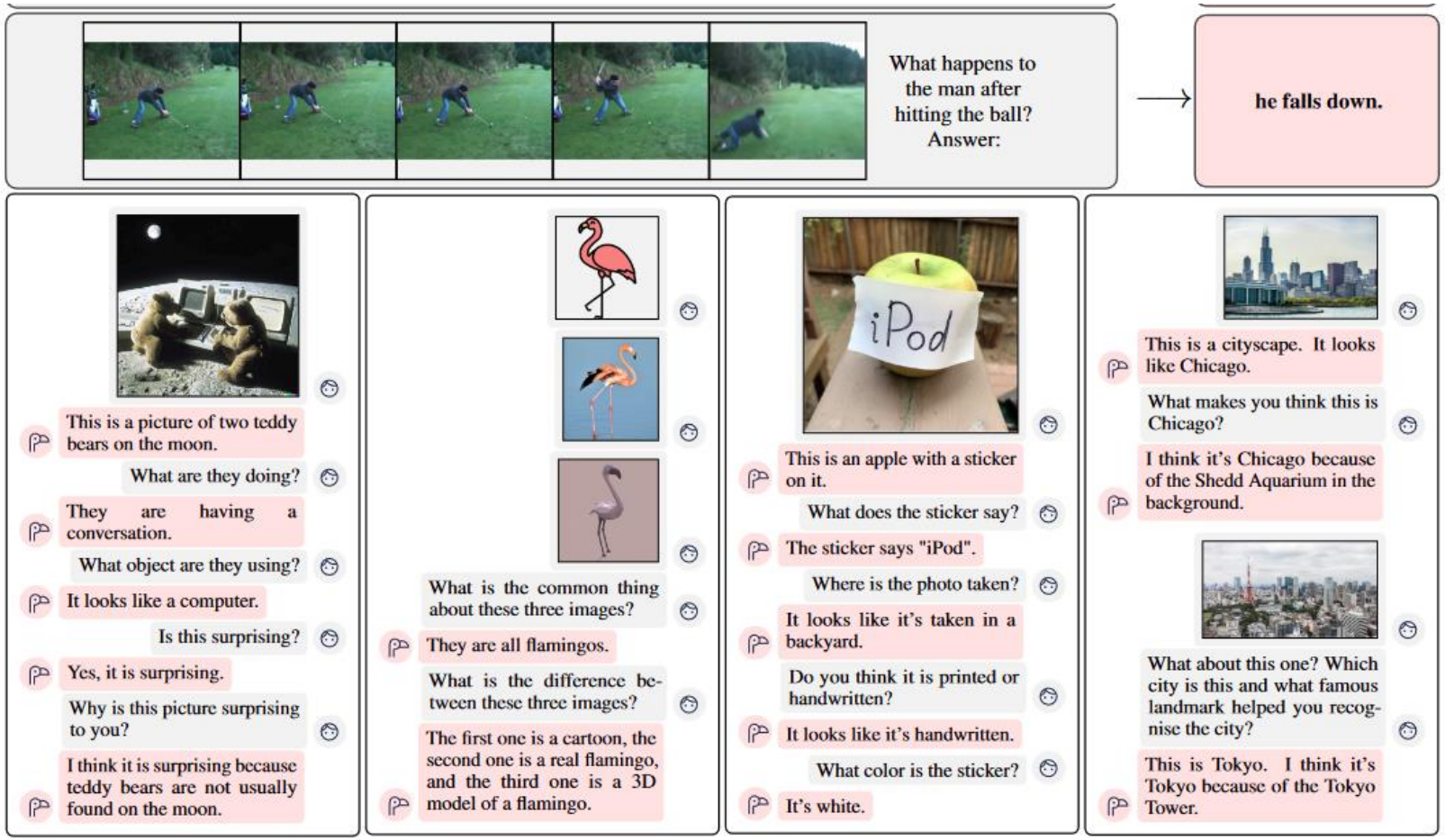



```
def gated_xattn_dense(
    y,  # input language features
    x,  # input visual features
    alpha_xattn, # xattn gating parameter – init at 0.
    alpha_dense, # ffw gating parameter – init at 0.
):
  """Applies a GATED XATTN-DENSE layer."""

  # 1. Gated Cross Attention
  y = y + tanh(alpha_xattn) * attention(q=y, kv=x)
  # 2. Gated Feed Forward (dense) Layer
  y = y + tanh(alpha_dense) * ffw(y)

  # Regular self-attention + FFW on language
  y = y + frozen_attention(q=y, kv=y)
  y = y + frozen_ffw(y)
  return y  # output visually informed language features
```

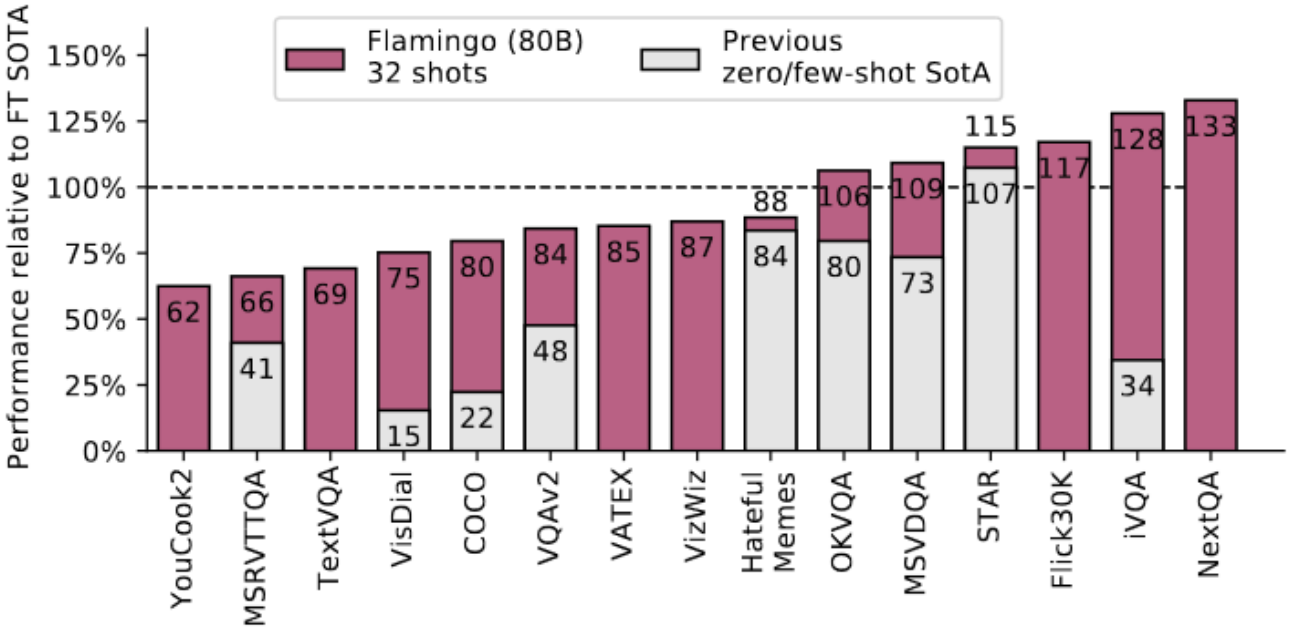
Flamingo

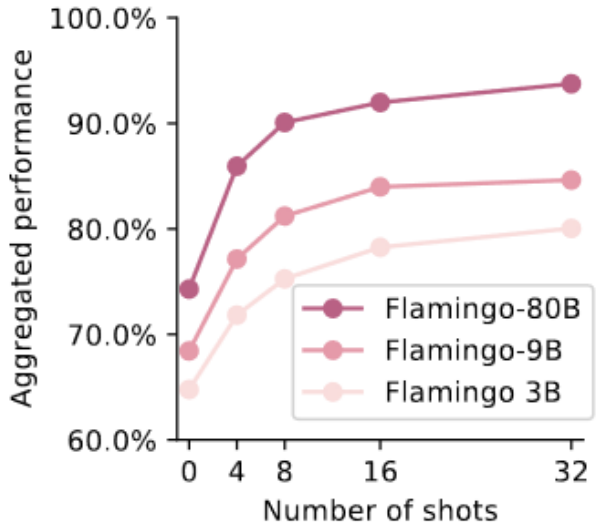

Input Prompt
Completion
This is a chinchilla. They are mainly found in Chile.
This is a shiba. They are very popular in Japan.
This is
a flamingo. They are found in the Caribbean and South America.
What is the title of this painting? Answer: The Hallucinogenic Toreador.
Where is this painting displayed? Answer: Louvres Museum, Paris.
What is the name of the city where this was painted? Answer:
Arles.
UNDERGROUND
Output: "Underground"
CONGRESS AVE
Output: "Congress"
SOULOMES
Output:
"Soulomes"

Flamingo


What happens to the man after hitting the ball? Answer:
he falls down.
This is a picture of two teddy bears on the moon.
What are they doing?
They are having a conversation.
What object are they using?
It looks like a computer.
Is this surprising?
Yes, it is surprising.
Why is this picture surprising to you?
I think it is surprising because teddy bears are not usually found on the moon.
What is the common thing about these three images?
They are all flamingos.
What is the difference between these three images?
The first one is a cartoon, the second one is a real flamingo, and the third one is a 3D model of a flamingo.
This is an apple with a sticker on it.
What does the sticker say?
The sticker says "iPod".
Where is the photo taken?
It looks like it's taken in a backyard.
Do you think it is printed or handwritten?
It looks like it's handwritten.
What color is the sticker?
It's white.
This is a cityscape. It looks like Chicago.
What makes you think this is Chicago?
I think it's Chicago because of the Shedd Aquarium in the background.
What about this one? Which city is this and what famous landmark helped you recognise the city?
This is Tokyo. I think it's Tokyo because of the Tokyo Tower.

Flamingo

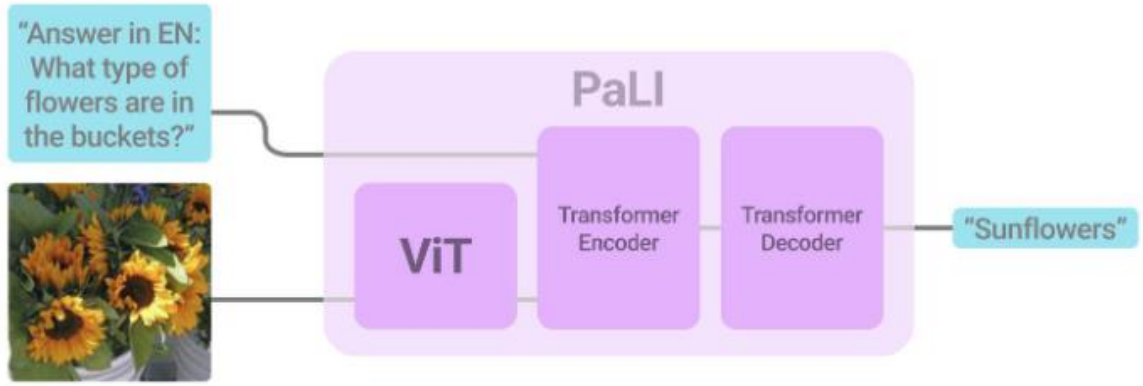

Flamingo (80B)
32 shots
Previous
zero/few-shot SotA
Performance relative to FT SOTA
150%
125%
100%
75%
50%
25%
0%
62
66
41
69
75
15
80
22
84
48
85
87
88
84
106
80
109
73
115
107
117
128
34
133
YouCook2
MSRVTTQA
TextVQA
VisDial
COCO
VQAv2
VATEX
VizWiz
Hateful Memes
OKVQA
MSVDQA
STAR
Flick30K
iVQA
NextQA


Aggregated performance
100.0%
90.0%
80.0%
70.0%
60.0%
0
4
8
16
32
Number of shots
Flamingo-80B
Flamingo-9B
Flamingo 3B

PaLI

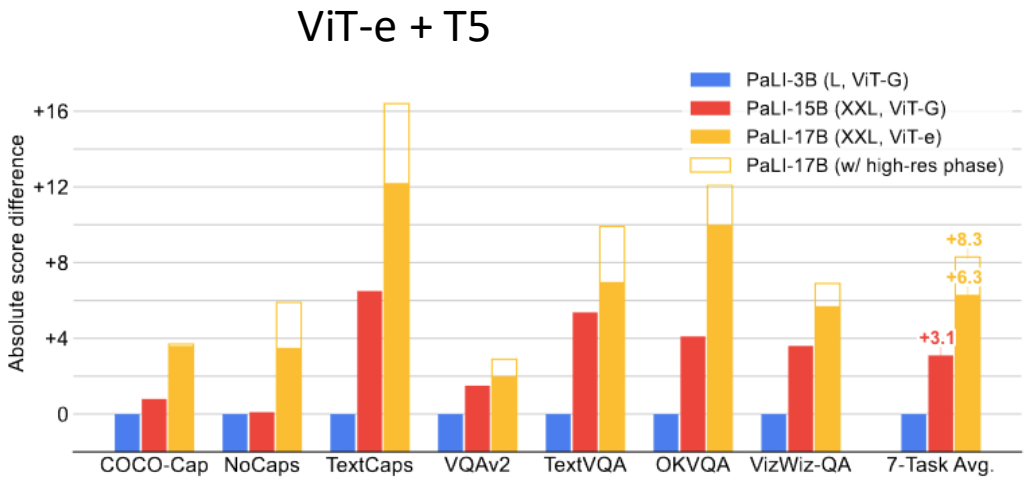
PaLI: A Jointly-Scaled Multilingual Language-Image Model

Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer Alexander Kolesnikov, Joan Puigcerver, Nan Ding, Keran Rong, Hassan Akbari Gaurav Mishra, Linting Xue, Ashish Thapliyal, James Bradbury, Weicheng Kuo Mojtaba Seyedhosseini, Chao Jia, Burcu Karagol Ayan, Carlos Riquelme Andreas Steiner, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, Radu Soricut
Google Research*



Abstract

Effective scaling and a flexible task interface enable large language models to excel at many tasks. We present **PaLI** (**Pa**thways **L**anguage and **I**mage model), a model that extends this approach to the joint modeling of language and vision. PaLI generates text based on visual and textual inputs, and with this interface performs many vision, language, and multimodal tasks, in many languages. To train PaLI, we make use of large pre-trained encoder-decoder language models and Vision Transformers (ViTs). This allows us to capitalize on their existing capabilities and leverage the substantial cost of training them. We find that joint scaling of the vision and language components is important. Since existing Transformers for language are much larger than their vision counterparts, we train a large, 4-billion parameter ViT (ViT-e) to quantify the benefits from even larger-capacity vision models. To train PaLI, we create a large multilingual mix of pre-training tasks, based on a new image-text training set containing 10B images and texts in over 100 languages. PaLI achieves state-of-the-art in multiple vision and language tasks (such as captioning, visual question-answering, scene-text understanding), while retaining a simple, modular, and scalable design.



PaLI Successors

PaLI-X

- Further scaling
- Better training schemes (mixture of objectives, further training with OCR objectives on vision encoder for better text understanding)

PaLI-3

- Switch to SigLIP encoder
- Close to PaLI-X while being 10x smaller (5B vs 55B)

Types

- Bidirectional Multi-Modal Models
- Encoder-Decoder Multi-Modal Models
- **Decoder-only Multi-Modal Models**
- Multi-Modal Generation

Frozen

Multimodal Few-Shot Learning with Frozen Language Models

Maria Tsimpoukelli[*]
DeepMind
mrts@deepmind.com

Jacob Menick[*]
DeepMind
University College London
jmenick@deepmind.com

Serkan Cabi[*]
DeepMind
cabi@deepmind.com

S. M. Ali Eslami
DeepMind
aeslami@deepmind.com

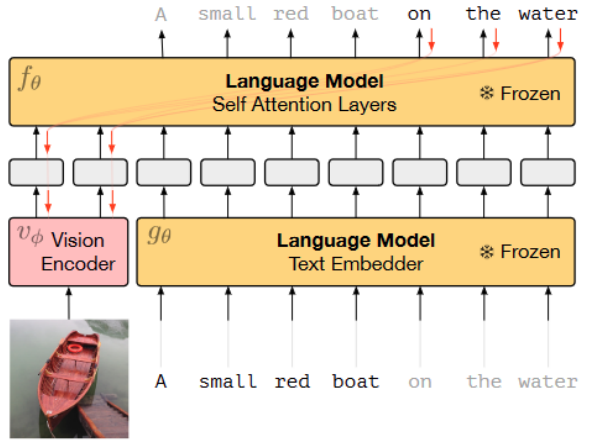
Oriol Vinyals
DeepMind
vinyals@deepmind.com

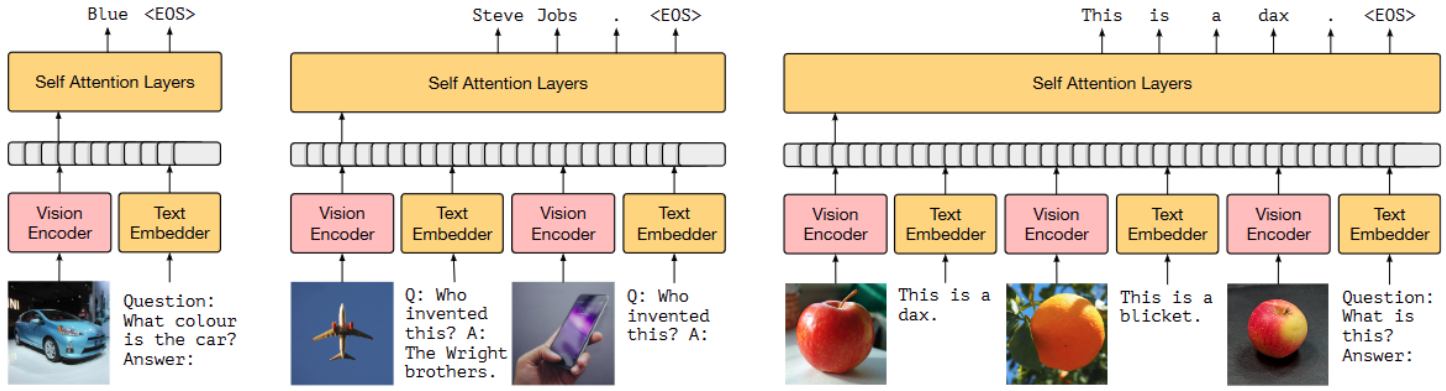
Felix Hill
DeepMind
felixhill@deepmind.com

Abstract

When trained at sufficient scale, auto-regressive language models exhibit the notable ability to learn a new language task after being prompted with just a few examples. Here, we present a simple, yet effective, approach for transferring this few-shot learning ability to a multimodal setting (vision and language). Using aligned image and caption data, we train a vision encoder to represent each image as a sequence of continuous embeddings, such that a pre-trained, frozen language model prompted with this prefix generates the appropriate caption. The resulting system is a multimodal few-shot learner, with the surprising ability to learn a variety of new tasks when conditioned on examples, represented as a sequence of multiple interleaved image and text embeddings. We demonstrate that it can rapidly learn words for new objects and novel visual categories, do visual question-answering with only a handful of examples, and make use of outside knowledge, by measuring a single model on a variety of established and new benchmarks.

NF-Resnet50 + 7B on C4





(a) **0-shot VQA** (b) **1-shot outside-knowledge VQA** (c) **Few-shot image classification**

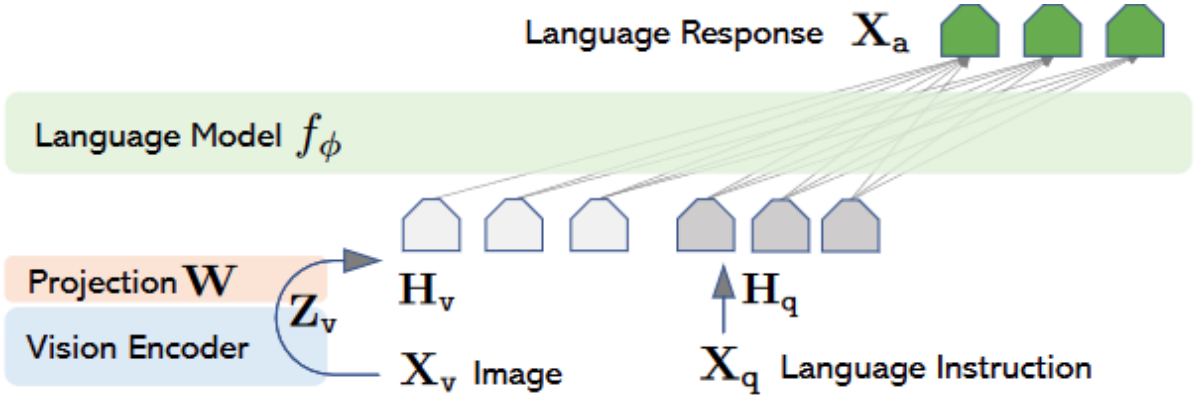
LLaVA

Visual Instruction Tuning

Haotian Liu[1*], **Chunyuan Li**[2*], **Qingyang Wu**[3], **Yong Jae Lee**[1]
[1]University of Wisconsin–Madison [2]Microsoft Research [3]Columbia University
https://llava-vl.github.io

Abstract

Instruction tuning large language models (LLMs) using machine-generated instruction-following data has been shown to improve zero-shot capabilities on new tasks, but the idea is less explored in the multimodal field. We present the first attempt to use language-only GPT-4 to generate multimodal language-image instruction-following data. By instruction tuning on such generated data, we introduce LLaVA: **L**arge **L**anguage **a**nd **V**ision **A**ssistant, an end-to-end trained large multimodal model that connects a vision encoder and an LLM for general-purpose visual and language understanding. To facilitate future research on visual instruction following, we construct two evaluation benchmarks with diverse and challenging application-oriented tasks. Our experiments show that LLaVA demonstrates impressive multimodal chat abilities, sometimes exhibiting the behaviors of multimodal GPT-4 on unseen images/instructions, and yields a 85.1% relative score compared with GPT-4 on a synthetic multimodal instruction-following dataset. When fine-tuned on Science QA, the synergy of LLaVA and GPT-4 achieves a new state-of-the-art accuracy of 92.53%. We make GPT-4 generated visual instruction tuning data, our model, and code publicly available.



Stage 1:

Keep vision + language backbones frozen, train projection layer on ~600K text-image pairs

Stage 2:

Keep vision backbone frozen, finetune language + projection on high quality visual instruct dataset

LLaVA

Challenging examples from LLaVA-Bench (In-the-Wild):		
	 ICHIRAN Ramen [source]	 Filled fridge [source]
Annotation	A close-up photo of a meal at ICHIRAN. The chashu ramen bowl with a spoon is placed in the center. The ramen is seasoned with chili sauce, chopped scallions, and served with two pieces of chashu. Chopsticks are placed to the right of the bowl, still in their paper wrap, not yet opened. The ramen is also served with nori on the left. On top, from left to right, the following sides are served: a bowl of orange spice (possibly garlic sauce), a plate of smoke-flavored stewed pork with chopped scallions, and a cup of matcha green tea.	An open refrigerator filled with a variety of food items. In the left part of the compartment, towards the front, there is a plastic box of strawberries with a small bag of baby carrots on top. Towards the back, there is a stack of sauce containers. In the middle part of the compartment, towards the front, there is a green plastic box, and there is an unidentified plastic bag placed on it. Towards the back, there is a carton of milk. In the right part of the compartment, towards the front, there is a box of blueberries with three yogurts stacked on top. The large bottle of yogurt is Fage non-fat yogurt, and one of the smaller cups is Fage blueberry yogurt. The brand and flavor of the other smaller cup are unknown. Towards the back, there is a container with an unknown content.
Question 1	What's the name of the restaurant?	What is the brand of the blueberry-flavored yogurt?
Question 2	Describe this photo in detail.	Is there strawberry-flavored yogurt in the fridge?

Types

- Bidirectional Multi-Modal Models
- Encoder-Decoder Multi-Modal Models
- Decoder-only Multi-Modal Models
- **Multi-Modal Generation**

Multi-modal Generation

Text->Image

- Stable Diffusion, Parti, Imagen

Text->Video

- ImagenVideo, Make-a-Video, Phenaki, Stable Video Diffusion

Text -> Video + Audio

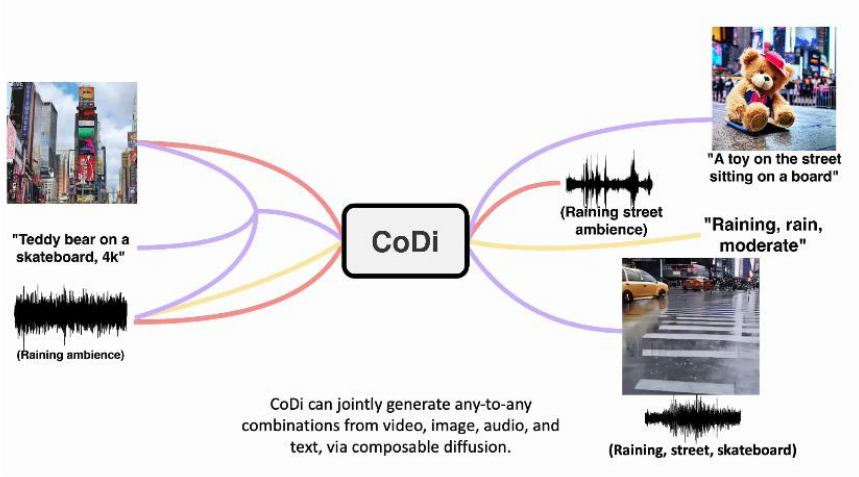
- VideoPoet

CoDi

Any-to-Any Generation via Composable Diffusion

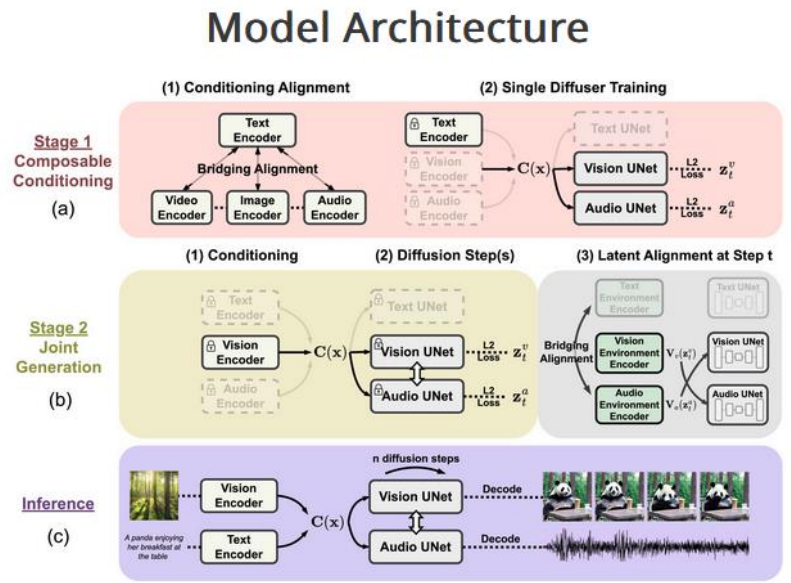
Zineng Tang[1]* **Ziyi Yang**[2]† **Chenguang Zhu**[2] **Michael Zeng**[2] **Mohit Bansal**[1]†
[1]University of North Carolina at Chapel Hill
[2]Microsoft Azure Cognitive Services Research

https://codi-gen.github.io



Abstract

We present Composable Diffusion (CoDi), a novel generative model capable of generating any combination of output modalities, such as language, image, video, or audio, from any combination of input modalities. Unlike existing generative AI systems, CoDi can generate multiple modalities in parallel and its input is not limited to a subset of modalities like text or image. Despite the absence of training datasets for many combinations of modalities, we propose to align modalities in both the input and output space. This allows CoDi to freely condition on any input combination and generate any group of modalities, even if they are not present in the training data. CoDi employs a novel composable generation strategy which involves building a shared multimodal space by bridging alignment in the diffusion process, enabling the synchronized generation of intertwined modalities, such as temporally aligned video and audio. Highly customizable and flexible, CoDi achieves strong joint-modality generation quality, and outperforms or is on par with the unimodal state-of-the-art for single-modality synthesis. The project page with demonstrations and code is at https://codi-gen.github.io/



Chameleon

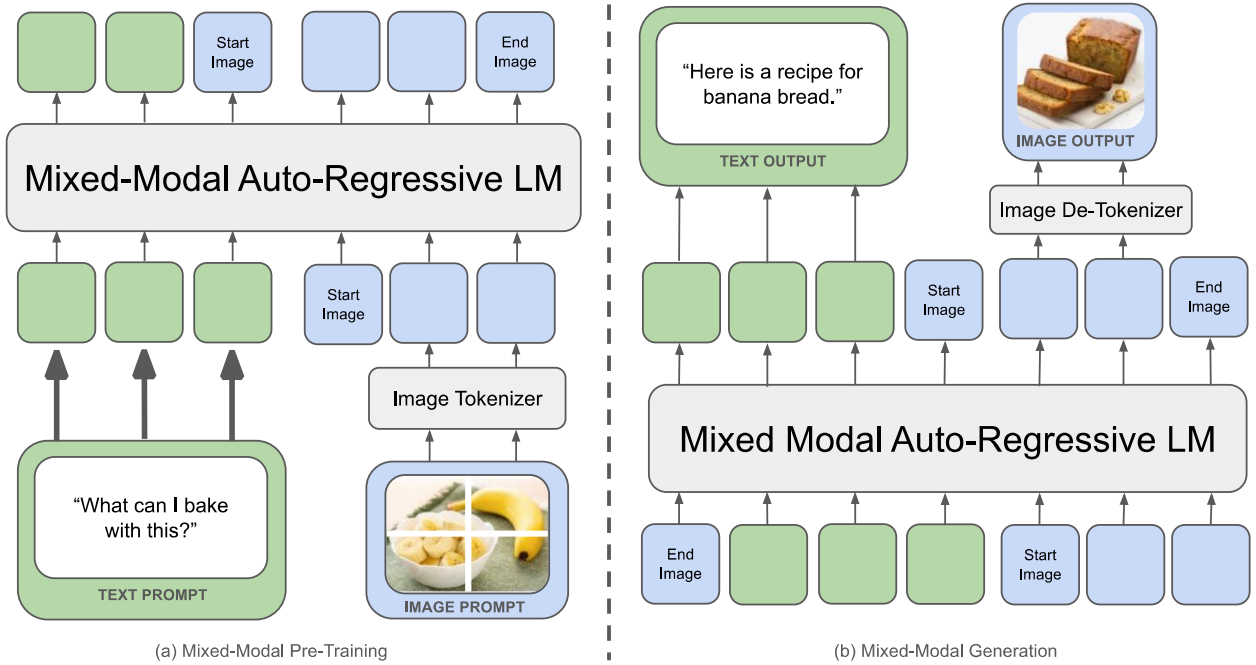
Chameleon: Mixed-Modal Early-Fusion Foundation Models

Chameleon Team[1,*]

[1]FAIR at Meta
[*]See Contributions section for full author list.

We present Chameleon, a family of early-fusion token-based mixed-modal models capable of understanding and generating images and text in any arbitrary sequence. We outline a stable training approach from inception, an alignment recipe, and an architectural parameterization tailored for the early-fusion, token-based, mixed-modal setting. The models are evaluated on a comprehensive range of tasks, including visual question answering, image captioning, text generation, image generation, and long-form mixed modal generation. Chameleon demonstrates broad and general capabilities, including state-of-the-art performance in image captioning tasks, outperforms Llama-2 in text-only tasks while being competitive with models such as Mixtral 8x7B and Gemini-Pro, and performs non-trivial image generation, all in a single model. It also matches or exceeds the performance of much larger models, including Gemini Pro and GPT-4V, according to human judgments on a new long-form mixed-modal generation evaluation, where either the prompt or outputs contain mixed sequences of both images and text. Chameleon marks a significant step forward in a unified modeling of full multimodal documents.

Date: May 17, 2024



(a) Mixed-Modal Pre-Training

(b) Mixed-Modal Generation

Transfusion

Transfusion: Predict the Next Token and Diffuse Images with One Multi-Modal Model

Chunting Zhou[μ*] **Lili Yu**[μ*] **Arun Babu**[δ†] **Kushal Tirumala**[μ]
Michihiro Yasunaga[μ] **Leonid Shamis**[μ] **Jacob Kahn**[μ] **Xuezhe Ma**[σ]
Luke Zettlemoyer[μ] **Omer Levy**[†]

[μ] Meta
[δ] Waymo [σ] University of Southern California

Abstract

We introduce Transfusion, a recipe for training a multi-modal model over discrete and continuous data. Transfusion combines the language modeling loss function (next token prediction) with diffusion to train a single transformer over mixed-modality sequences. We pretrain multiple Transfusion models up to 7B parameters from scratch on a mixture of text and image data, establishing scaling laws with respect to a variety of uni- and cross-modal benchmarks. Our experiments show that Transfusion scales significantly better than quantizing images and training a language model over discrete image tokens. By introducing modality-specific encoding and decoding layers, we can further improve the performance of Transfusion models, and even compress each image to just 16 patches. We further demonstrate that scaling our Transfusion recipe to 7B parameters and 2T multi-modal tokens produces a model that can generate images and text on a par with similar scale diffusion models and language models, reaping the benefits of both worlds.

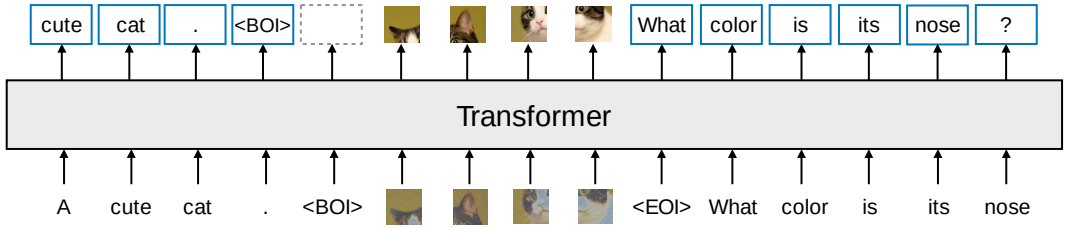
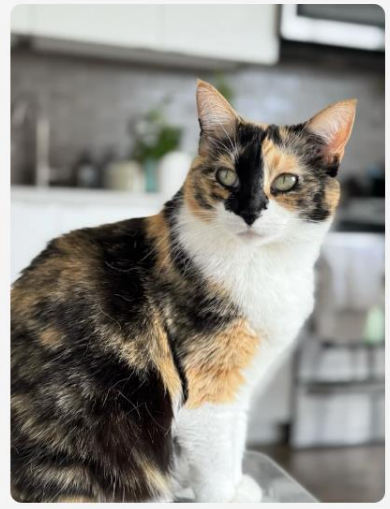
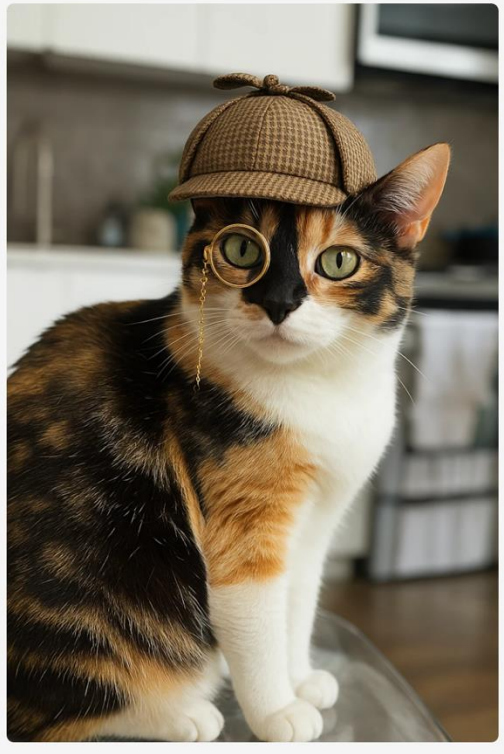


Figure 1: A high-level illustration of Transfusion. A single transformer perceives, processes, and produces data of every modality. Discrete (text) tokens are processed autoregressively and trained on the next token prediction objective. Continuous (image) vectors are processed together in parallel and trained on the diffusion objective. Marker BOI and EOI tokens separate the modalities.



Give this cat a detective hat and a monocle



turn this into a triple A video games made with a 4k game engine and add some User interface as overlay from a mystery RPG where we can see a health bar and a minimap at the top as well as spells at the bottom with consistent and iconography

Best of 1

update to a landscape image 16:9 ratio, add more spells in the UI, and unzoom the visual so that we see the cat in a third person view walking through a steampunk manhattan creating beautiful contrast and lighting like in the best triple A game, with cool-toned colors

Best of 2